ACCEPTED FROM OPEN CALL

AN ALOHA-BASED IMPROVED ANTI-COLLISION ALGORITHM FOR RFID SYSTEMS

YEJUN HE AND XIAOYE WANG, SHENZHEN UNIVERSITY

ABSTRACT

Tag collision arbitration is considered as one of the critical issues in RFID system design. In order to further improve the identification efficiency of tag anti-collision algorithms in RFID systems, several types of dynamic framed slotted ALOHA (DFSA) anti-collision algorithms are analyzed, and a new anti-collision algorithm is proposed. The proposed algorithm has the ability of identifying the time slot distribution which is selected by the tags within a reader's interrogation range in advance. Then the free time slots will be skipped when the reader queries each time slot. Also, the colliding tags will be immediately processed with additional time slots. Simulation results show that the proposed algorithm takes fewer total number of time slots and has a higher efficiency of tag identification compared to the other four DFSA anti-collision algorithms.

INTRODUCTION

Radio frequency identification (RFID) is a contactless automatic identification technology, which uses radio frequency signals to identify specific objects. Compared with bar codes, magnetic cards or IC card, RFID has many advantages such as long recognition distance, no line-of-sight requirement, simultaneous multi-target identification, resistance to interference. A typical RFID system is composed of multiple tags and a reader [1, 2]. The reader sends a command to the tags or receives related signals from the tags; while each tag is an electronic device attached to the object to be identified. The reader identifies the tags through a shared wireless communication channel at a certain carrier frequency[3, 4]. Suppose there are two or more tags in the interrogation zone of a reader and the reader broadcasts a query command. If all the tags that receive the message are to send their responses back to the reader at the same time, the responses will collide and the reader will not be able to identify the tags correctly. Such an event is referred to as a "collision," which is an important issue in an RFID system. It affects (i) the time taken to identify all the tags and (ii) the accuracy of the identification. Consequently, various anti-collision algorithms have been proposed and they can be classified into two categories: tree-based and ALOHA-based.

Tree-based algorithms [5] resolve a collision by splitting colliding tags into subsets iteratively until all the tags are identified. Such algorithms can identify all the tags but they are computationally complex. Moreover, they require relatively long identification delays, especially when the number of tags is large. Meanwhile, ALOHA-based algorithms reduce the probability of tag collision by dividing the time into discrete time intervals (time slots) and sequentially identifying the tags at different time slots [6]. Such algorithms can be implemented easily and used with an arbitrary number of tags [7]. Hence, ALOHA-based algorithms are the most prevalent ones used in the ultra-high-frequency (UHF) band in RFID systems. In particular, the dynamic framed slotted ALOHA (DFSA) algorithms are the most popular [8–12].

In the DFSA algorithms, the number of tags is usually estimated so as to determine frame size and to maximize the tag identification efficiency. For example, the DFSA algorithms proposed in [8, 9] make use of collision probability to estimate the number of tags. Schoute [10] thinks that the number of colliding tags is equal to 2.39 times the number of collided time slots. The number of tags in a low bound method proposed by Vogt is $S + 2C$ [11], where S is the number of single occupied time slots and C is the number of collision time slots. If the number of tags is larger than the initial frame size, a reader will not be able to estimate the number of tags exactly because the probability of collision becomes larger. On the other hand, if the number of tags is smaller than the initial frame size, many time slots in the frame will be idle. This leads to a long identification time and a low efficiency. Moreover, a DFSA algorithm based on the Q-slot algorithm is proposed in the ISO/IEC 18000-6 Type C standard [1]. The use of the Q-slot algorithm is to determine the frame size that can maximize the tag identification efficiency. In this algorithm, the value of Q is updated slot by slot according to the status of the preceding received slot. Splitting BTSA (binary tree slotted ALOHA) [13] is the latest proposed anti-collision algorithm, where a splitting method is utilized to adjust the frame size to a value close to the number of tags. It is also clear from the literatures that most of the DFSA anti-collision algorithms have one fea-

1536-1284/13/$25.00 © 2013 IEEE

ture in common — they identify all the tags by setting a proper frame size iteratively and hence reducing the probability of tag collision. However, they do not deal with the colliding tags immediately and the idle time slots are wasted.

In this article, we present an improved anti-collision algorithm based on the DFSA anti-collision algorithm. The proposed algorithm has two objectives. The first objective is to eliminate the idle time slots in a frame. Such an action can reduce the identification time and enhance the system efficiency. The other objective is to immediately process the colliding tags in a collision time slot. We propose allocating extra time slots to the colliding tags so as to reduce the probability that the tags collide again in the future.

The remainder of the article is organized as follows. First, we provide an overview and points out the shortcomings of the DFSA anti-collision algorithm. The next section describes our proposed anti-collision algorithm. The simulation results and performance analysis for the proposed algorithm are then given, and the final section concludes the article.

Related Work

Dynamic Framed Slotted ALOHA

In the DFSA algorithms, time is divided into a series of discrete time intervals called *time slots*. Moreover, several time slots are packaged to form a frame [8, 9]. Figure 1 illustrates the structure of a frame consisting of F time slots. Upon receiving a query command that specifies a frame size of (F) from a reader, the tag will randomly select a time slot between 1 and F to send its ID information. Details of the DFSA algorithm [9] are as follows:

Step 1: The reader broadcasts a query command which contains the parameter frame size (F).

Step 2: Each of the tags located within the reader's interrogation zone randomly select a time slot between 1 and F and transmits its ID at the selected time slot.

Step 3: The reader attempts to read a tag ID in each of the time slots. Within each time slot, there are only three possible outcomes: no ID is sent, only one ID is sent, and multiple IDs are sent. When no ID is sent in a time slot, the time slot becomes *idle* and the resource is wasted. When only one ID is sent in a time slot, the transmission is *successful* because the reader will be able to identify the ID. A collision will happen when multiple tags send their IDs in the same time slot. The transmissions *collide* and the reader will not be able to identify the IDs.

Step 4: At the end of each frame, the reader records the numbers of idle time slots, successful time slots, and collided time slots. If there are collided time slots, the reader will estimate the number of unidentified tags in the interrogation zone. Then an appropriate number of time slots is determined for the following frame. The identification process will stop when there is no collided time slots in the read process, i.e., all IDs have been identified.

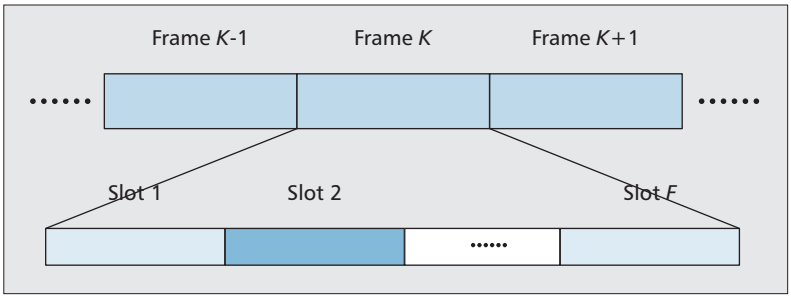

Figure 1. *Structure of the frame.*

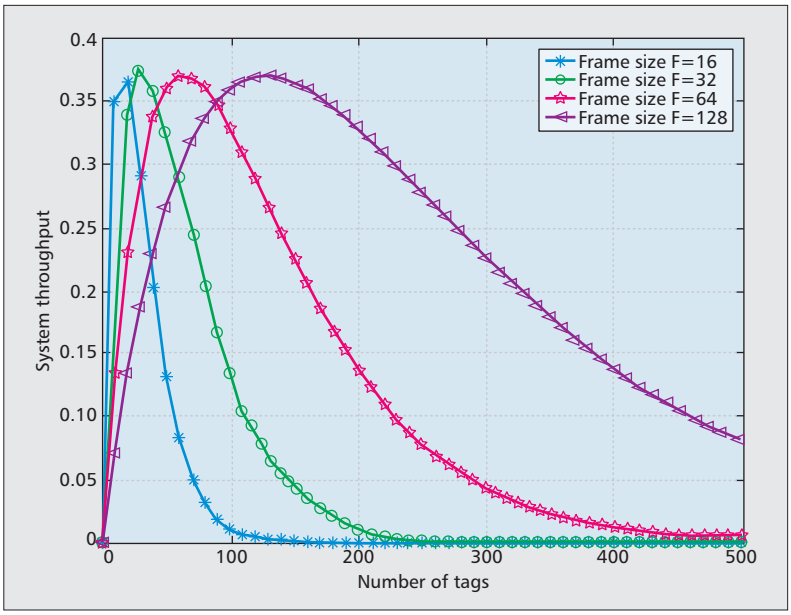

Figure 2. *System throughput vs. number of tags.*

For a particular frame, the throughput [9] is determined by the frame size (i.e., number of time slots in the frame) and the number of tags in the interrogation zone. Figure 2 shows the relationship between the throughput and the number of tags in the various frame size F. From Fig. 2, we can see that the maximum throughput is attained when the frame size is equal to the number of tags.

Drawback of Existing DFSA Anti-Collision Methods

In the DFSA anti-collision algorithms, a tag estimation method is employed to estimate the number of remaining tags at the end of each frame. Then the size of the next frame is adjusted accordingly. No further process is performed for the idle time slots and collided time slots in the current frame. Therefore, the idle time slots and the corresponding request commands sent by the reader during these time slots are wasted. Also, the reader does not immediately process the tag IDs sent in the collided time slots, but attempts to identify all these collided tag IDs in the next frame. Consequently, tag collision may also occur next time for these tags. In the next section, an improved algorithm is proposed. The algorithm aims at determining the selected time

For each of the selected time slots, if a collision occurs, the reader will roughly determine the number of extra time slots required to resolve the colliding tags. In our algorithm, we make the decision based on the ratio of the number of time slots selected to the frame size.

slots in advance, skipping the idle time slots and instantly processing the collided time slot by the reader.

The Improved Algorithm Based on DFSA

Description of the Proposed Algorithm

In this section, an improved anti-collision algorithm based on DFSA is proposed. It has the ability

- To find out in advance the time slot numbers selected by the tags
- To attempt resolving the tag ID collisions when they occur

In our algorithm, it is assumed that the number of tags is approximately equal to the initial frame size.

For a given time slot number, the reader firstly analyzes whether it is a selected time slot or not. Then the reader forms a frame with only the time slots that have been selceted. In other words, all the non-selected time slots have been removed and only the selected ones remain in the frame. Such an action can greatly improve the efficiency of the RFID system because no time is now spent on the time slots not selected by the tags.

Next, for each of the selected time slots, if a collision occurs, the reader will roughly determine the number of extra time slots required to resolve the colliding tags. In our algorithm, we make the decision based on the ratio of the number of time slots selected to the frame size (in time slot). If this ratio is less than 0.5, many time slots are not selected. Hence there is a relatively large probability that the collided time slot has been selected by 3 or 4 tags. Consequently, we append 4 additional time slots to the frame for the colliding tags to select. However, if the ratio is larger than 0.5, there is also a good chance that the collided time slot has been selected by two tags. In such a case, we append 2 additional time slots to the frame for the colliding tags to select.[1]

In order to implement the proposed algorithm, the following notations are adopted:

- F: An initial frame size on the reader side.
- *Sel_slot*: The total number of time slots that are selected by the tags in a reader's interrogation zone on the reader side.
- *Ratio_slot*: The ratio of the number of selected time slots to the frame size (in time slot) on the reader side.
- *Position_freeslot*(m, n): The position of non-selected time slots in a frame on the reader side. Here, m denotes the time slot number of the first non-selected time slot, and n denotes the number of non-selected time slots that follow.
- *Add_F*: The additional time slots used for identifing colliding tags on the reader side.
- M: A random number generator on the tag side.
- C: A counter on the tag side.

The proposed algorithm is executed as follows.

Step 1: The reader broadcasts the query command "Query(Q)" to start the procedure of tag identification. The command indicates that the frame size is equal to 2^Q.

Step 2: Each of the tags in the reader's interrogation zone randomly selects a time slot number between 1 and 2^Q, and loads the selected time slot number into C.

Step 3: All tags send their selected time slot numbers to the reader at the same time using sequences with a length of 2^Q bits. Denoting the time slot number selected by a particular tag by k, the sequence is then an all-zero sequence except at the k-bit location where it is a one. For example, if the frame size is 8 and a tag selects the time slot number 2, the corresponding sequence becomes "01000000." Furthermore, during the transmission, the sequence is coded by the Manchester code. Note that Manchester code technique can synchronize all tag transmissions at the beginning of each communication session.

Step 4: Upon receiving the sequences from all the involving tags, the reader can easily recognize time slots that are selected and those that are not selected by the tags. At the same time, the ratio of the number of selected time slots to the frame size (i.e., *Ratio_slot*) can be computed.

Step 5: The reader jumps to the next time slot that has been selected by the tags. In other words, all non-selected time slots are skipped. The reader then sends a query command for this time slot (i.e., "QueryRep(*slot*)") to all the tags and waits for the response. Two scenarios may occur. The first one is that only one tag responds and its ID can be successfully identified by the reader. The other scenario is that two or more tags respond and a collision arises.

Step 6: If a collision occurs, the reader will transmit the command "QueryAdd(*Add_F*)" to provide additional time slots (append to the frame) for the colliding tags to select. The colliding tags send their selected time slot numbers to the reader again and the reader updates the numbers of the selected time slots.

Step 7: Repeat steps 5 and 6 until all tags are successfully identified.

Flowchart of the Proposed Algorithm

In the proposed algorithm, the following new commands are used:

- "Query(Q)": The reader broadcasts a command to start a new read cycle. The frame size is equal to 2^Q.
- "QueryRep(*slot*)": The reader queries a certain time slot *slot*.
- "QueryAdd(*Add_F*)": This command is used to deal with the colliding tag IDs. The reader sends this command to let all colliding tags to adjust their frame size.

In Fig. 3, we show the flowchart of our proposed algorithm.

The detailed description is as follows:

1. A reader broadcasts a query command "Query(Q)" with value Q to all tags within reading range.
2. Upon receiving the command "Query(Q)," each of the tags randomly selects a time slot between 1 and 2^Q and transmits its ID. Then it sends the time slot number selected back to the reader at the same time in the form of "only one sequence."

[1] *In theory, the number of tags selecting a particular time slot can be computed based on binomial distribution theory or approximated by Poisson distributions [12. Given the frame size and the number of non-selected time slots, we should be able to estimate the probability that a timelsot is selected by k tags, where* k ≥ *1. Based on such information, we can more precisely determine the number of additional time slots that we should assign for the colliding tags to select. Here, we only use a simple decision.*

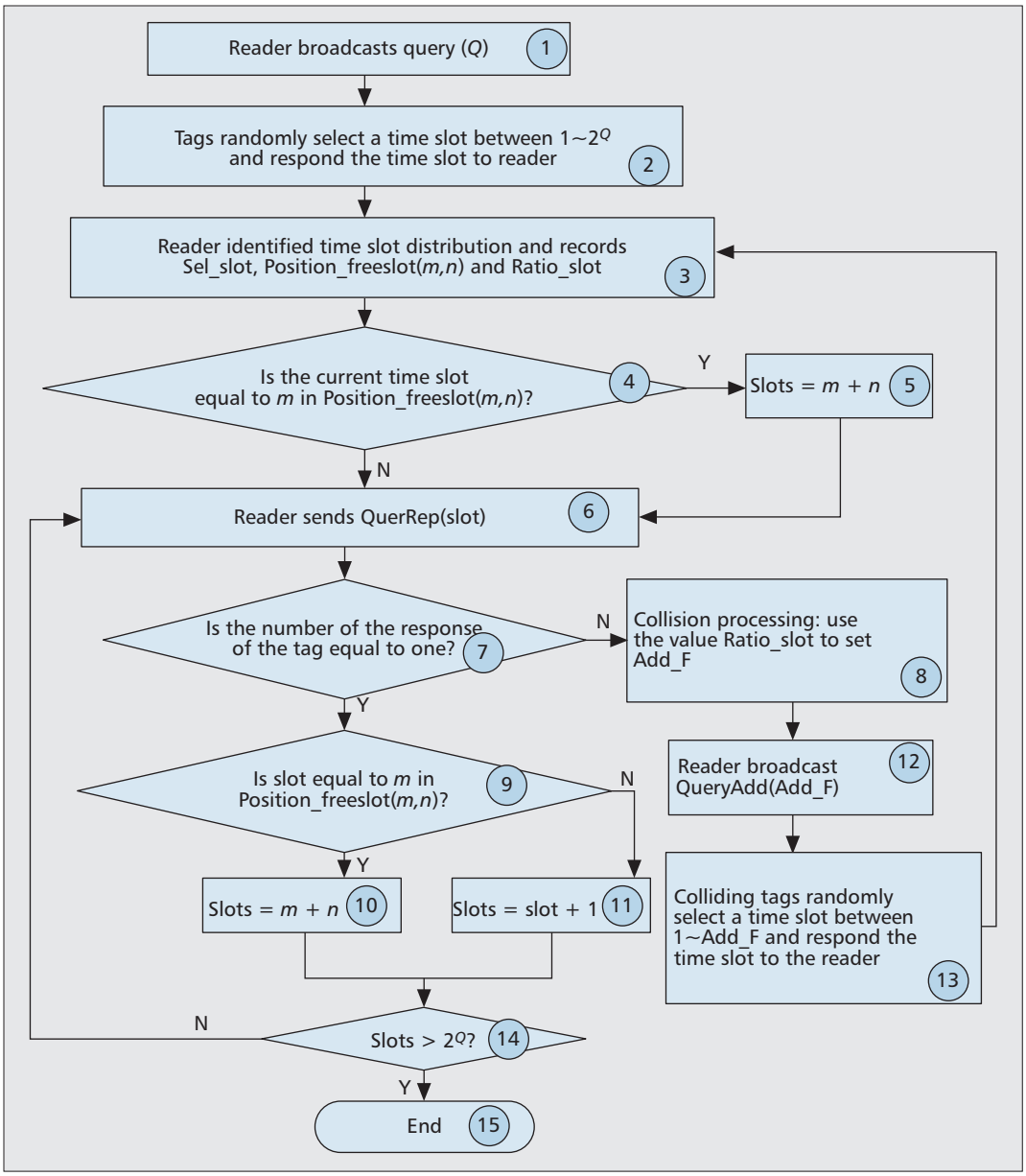

Figure 3. *Flowchart of the proposed architecture.*

3. After getting the sequences from all tags, the reader calculates the total number of selected time slots and the positions of the idle time slots in one frame. Also, the ratio of the total number of selected time slots to the frame size is computed. Let $slot = 0$.
4. Before querying a time slot, the reader needs to determine whether the queried time slot is an idle time slot. If yes, the system goes to 5. Otherwise, it goes to 6.
5. The current time slot *slot* is adjusted to $m + n$.
6. The reader starts to query the time slots one by one by transmitting the command "QuerySlot(*slot*)," where slot denotes a time slot number (i.e., current time slot).
7. The reader waits for the response of the tag. Since all idle time slots have been skipped, there are only two possible outcomes for a given time slot. Either one tag responds or multiple tags respond, and the latter case will result in tag collision.
8. If collision occurs, the system will start collision identification procedure immediately. The number of colliding tags will be estimated based on the value *Ratio_slot* and a new frame size *Add_F* is allocated to 2 or 4 based on *Ratio_slot*.
9. The tag will be successfully identified when only one tag responds. Then the current time slot is compared with the value m in *Position_freeslot*(m, n). If the current time slot is equal to m, the system goes to 10. Otherwise, the system goes to 11.
10. The next query time slot number is adjusted to $m + n$.
11. Increment the current time slot by 1.
12. The reader sends "QueryAdd(*Add_F*)" to the colliding tags and waits for the responses of the colliding tags.
13. Upon receiving the command "QueryAdd(*Add_F*)," each of the colliding tags randomly selects a time slot between 1

The total time to identify all tags is equal to the total number of time slots multiplied by the slot time. Since the slot time is constant, we only take the total number of time slots into consideration. The smaller the total number of time slots is, the better the performance of the algorithm is.

```
Input : number of tags
Output : total time slot and efficiency of tag identification
    ----------Reader Side----------
  Initial: Q, F, Sel_slot, Ratio_slot, Position_freeslot(m,n)
         and Add_F
  Broadcast : Query(Q)
  Get the responses of time slot from all tags
Analyze the response of time slot information
  Record Sel_slot, Position_freeslot(m,n)
          and compute Ratio_slot
Query time slot one by one
   m = 0, k = 0
   while k <= F
       if k == position_freeslot(m,l)  {
          k = k + position_freeslot(m,2);
          m = m + 1;
          continue;  }
       else if successful time slot
                 total_slot++;
        k++;
       else collision time slot
                 total_slot++;
   go to Used_slot = collision_proc(Ratio_slot, k);
             total_slot = total_slot + Used_slot;
   end
 end
 Broadcast end round message
   ----------Tag Side----------
 Initialize C = 0, OnlyO_sequence = 0
 Respond Query(Q) : OnlyO_sequence=random(1~2^Q), C =
           Only O_sequence and transmit the Only O-sequence
 While C > 0
   If Receive QueryRep(slot): C = C – 1
   end if
  end while
```

Table 1. *Pseudo codes of the proposed algorithm.*

and Add_F and transmits its ID. Then it sends the time slot number selected back to the reader at the same time in the form of "only one sequence."

14. If the current time slot is less than 2^Q, the system goes to 6. Otherwise, the system goes to 15.
15. The identification procedure ends.

The pseudo code of our proposed algorithm is futher depicted in Table 1.

Simulation Results and Analysis

We evaluate the performance of the proposed algorithm and compare it with that of the Schoute DFSA algorithm [10], low bound DFSA [11], Q-algorithm (i.e., the DFSA algorithm in ISO/IEC 18000-6 Type C [1]), and splitting BTSA [13]. In our simulations, the number of tags ranges from 5 to 1000. Moreover, for each setting, the result is the average of 100 simulations.

Since the performance of a RFID anti-collision algorithm is mainly determined by the efficiency of tag identification and the total time taken to identify all the tags, these two factors are taken into consideration in our simulations.

First, the efficiency of tag identification, denoted by ρ, is measured by the ratio of the number of tags (N_{tags}) to the total number of time slots (N_{slots}), that is,

$$\rho = \frac{N_{tags}}{N_{slots}} \quad (1)$$

A larger ρ indicates an algorithm with a higher performance. The total time to identify all tags is equal to the total number of time slots multiplied by the slot time. Since the slot time is constant, we only take the total number of time slots into consideration. The smaller the total number of time slots is, the better the performance of the algorithm is.

Figure 4a plots the total number of time slots needed to identify all the tags against the number of tags. The number of tags increases from 5 to 100. In Schoute DFSA, low bound DFSA and Q-algorithm, the initial frame size is set to 16 time slots. Moreover, the initial frame size in our proposed algorithm is set to 64 time slots. From Fig. 4a, we can see that the performance of the Schoute DFSA, low bound DFSA and Q-algorithm are very close because they adjust the frame size based on estimating the number of tags. The splitting BTSA, which utilizing a splitting method to set the frame size to a value close to the number of tags, provides a better performance than the above three algorithms. Moreover, our proposed algorithm requires a smaller number of time slots compared with other algorithms and hence achieves the best performance. Figure 4b plots the total number of time slots vs. the number of tags for the algorithms when the number of tags ranges from 100 to 1000. The initial frame size is now set to 256 time slots in our proposed algorithm because of the large number of tags. In Schoute DFSA, low bound DFSA and Q-algorithm, the initial frame size is set to 16. Again, as the number of tags increases, the total number of time slots increases quickly due to tag collisions. However, our proposed algorithm gives the best performance in terms of the total number of time slots, especially for a large number of tags.

Figure 5a further plots the efficiency of tag identification for the different algorithms. We can observe that the efficiency of each of the Schoute DFSA, low bound DFSA, Q-algorithm and spliting BTSA becomes very stable when the number of tags is beyond 50. Among these algorithms, moreover, the splitting BTSA attains a higher efficiency. However, our proposed algorithm outperform other algorithms and achieves the highest efficiency. We can also see that the efficiency of our proposed algorithm decreases with the number of tags. The reason is that when there are more tags, more collisions occur and more additional time slots are required to resolve such colliding tags. Figure 5b further shows that the efficiency of tag identification for the algorithms. When the number of tags is 100, the efficiency of tag identification of the proposed algorithm is about 82 percent while that of the other algorithms ranges between 30 and 45 percent. There exists a performance gap between the proposed algorithm and the other four simulated algorithms because different methods are used to deal with the collision of tags. With the increasing of the number of tags, tag collisions increase and additional time slots are required to identify the colliding tags. For a

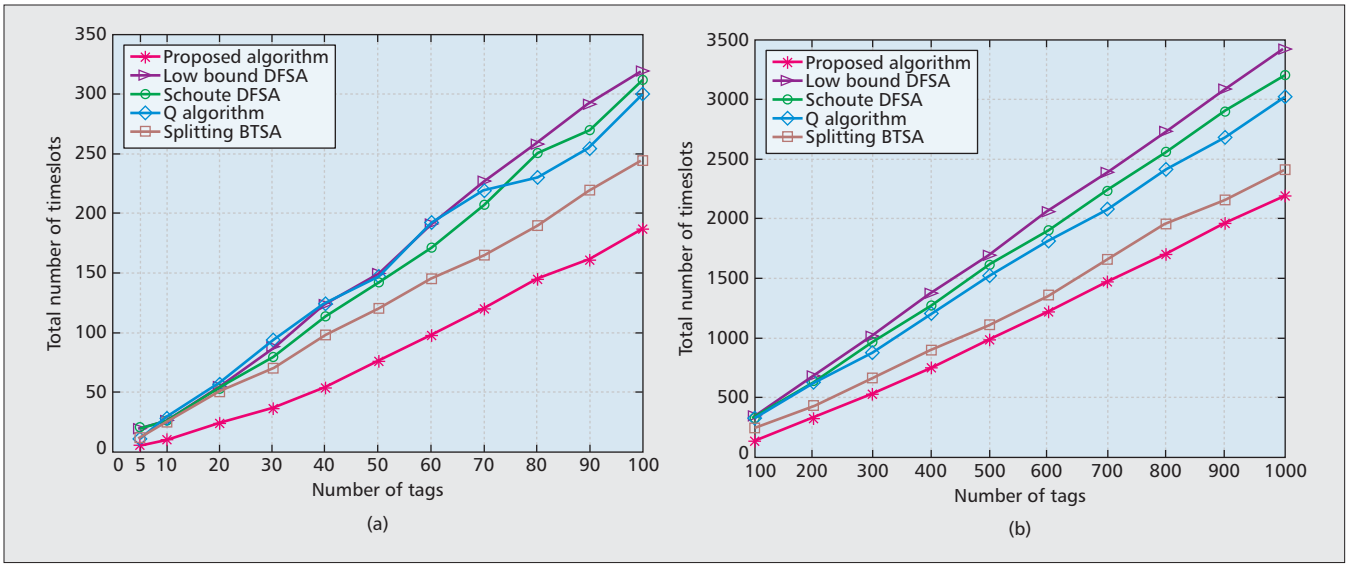

Figure 4. *Total number of time slots vs. number of tags.*

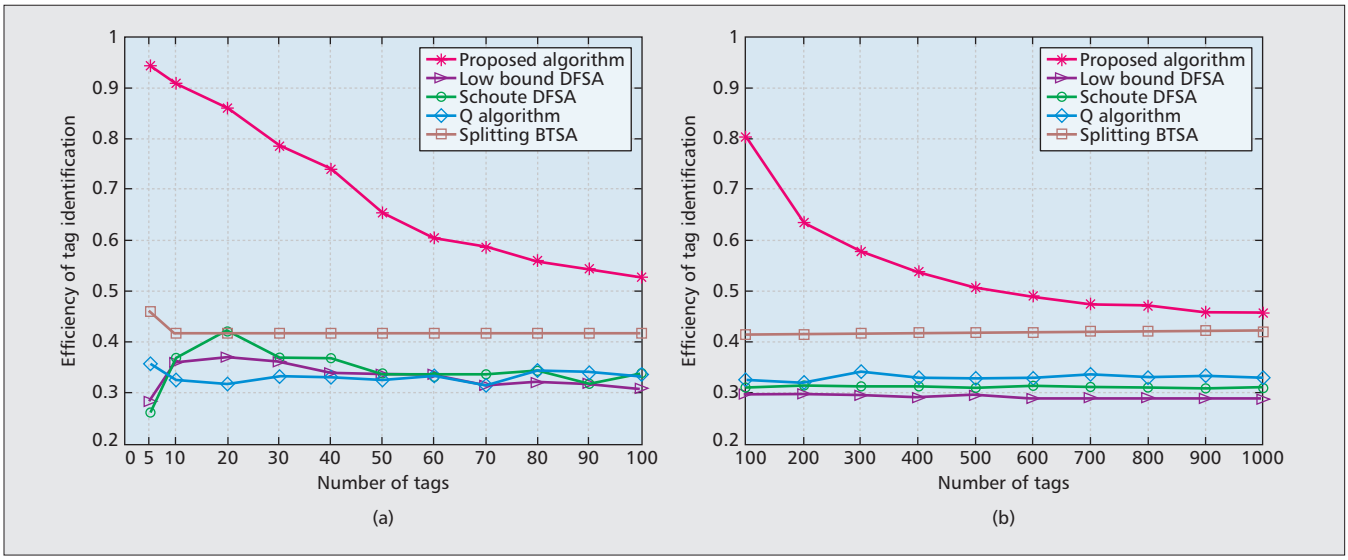

Figure 5. *Efficiency of tag identification vs. number of tags.*

large number of tags, the efficiency of the proposed algorithm is above 45 percent while that of the splitting BTSA is about 43 percent and that of the other three algorithms is only about 30 percent. Thus, the performance of the proposed algorithm is the best.

Conclusion

In this article, an improved algorithm based on the dynamic framed slotted ALOHA anti-collision algorithm was proposed. We analyzed the position of idle time slots in a frame. The idle time slots were skipped when the reader queried tags. For a collision time slot, a collision procedure was instantly started and additional time slots were set to deal with the colliding tags. Simulation results show that total number of time slots of our proposed algorithm was much fewer than that of the splitting BTSA, low bound DFSA, Schoute DFSA and Q-algorithm in ISO/IEC 18000-6 Type C. Also, the efficiency of tag identification was greatly improved. Therefore, the identification performance of the proposed scheme is significantly improved.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (No. 60972037 and No. 61372077), the Fundamental Research Program of Shenzhen City (No. JC201005250067A and No. JCYJ20120817163755061), the International Cooperative Program of Shenzhen City (No. ZYA201106090040A), and the Joint Industry-Teaching-Research Program of Guangdong Province and the Ministry of Education (No. 2011B090400512). We would like to thank Prof. Francis C.M. Lau of the Hong Kong Polytechnic University for his valuable suggestions. We would also like to thank the Editor and the anonymous reviewers for their insights that improved the article significantly.

For a large number of tags, the efficiency of the proposed algorithm is above 45 percent while that of the splitting BTSA is about 43 percent and that of the other three algorithms is only about 30 percent. Thus, the performance of the proposed algorithm is the best.

References

[1] EPCglobal Standard Spec., "EPC™ Radio-Frequency Identification Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz–960 MHz ver. 1.0.9.," Jan. 2005, pp. 1–94.

[2] R. Want, "An Introduction to RFID Technology," *IEEE Pervasive Computing*, vol. 5, no. 1, Jan.–Mar. 2006, pp. 25–33.

[3] Z. Shi, C. Beard, and K. Mitchell, "Analytical Models for Understanding Misbehavior and MAC Friendliness in CSMA Networks," *Performance Evaluation*, vol. 66, Sept. 2009, pp. 469–87.

[4] Z. Shi, C. Beard, and K. Mitchell, "Analytical Models for Understanding Space, Backoff and Flow Correlation in CSMA Wireless Networks," *Wireless Networks*, Springer, published online, July 2012.

[5] Z. Yu and X. Liu, "Improvement of Dynamic Binary Search Algorithm Used in RFID System," *Cross Strait Quad-Regional Radio Science and Wireless Technology Conf.*, vol. 2, July 2011, pp. 1046–49.

[6] L. Zhu and T. S. P. Yum, "A Critical Survey and Analysis of RFID Anti-Collision Mechanisms," *IEEE Commun. Mag.*, vol. 49, no. 5, May 2011, pp. 214–21.

[7] S. Piramuthu, "Anti-Collision Algorithm for RFID Tags," *Proc. Conf. Mobile and Pervasive Computing*, Aug. 2008, pp. 116–18.

[8] S. R. Lee, S. D. Joo, and C. W. Lee, "An Enhanced Dynamic Framed Slotted ALOHA Algorithm for RFID Tag Identification," *2nd Annual Int'l. Conf. Mobile and Ubiquitous Sys.: Networking and Services*, July 2005, pp. 166–72.

[9] W. T. Chen, "An Accurate Tag Estimate Method for Improving the Performance of an RFID Anticollision Algorithm Based on Dynamic Frame Length Aloha," *IEEE Trans. Automation Sci. and Eng.*, vol. 6, no.1, Jan. 2009, pp. 9–15.

[10] F. C. Schoute, "Dynamic Frame Length ALOHA," *IEEE Trans. Commun.*, vol. 31, no. 4, Apr. 1983, pp. 565–68.

[11] H. Vogt, "Efficient Object Identification with Passive RFID Tags," *Proc. Int'l. Conf. Pervasive Computing, LNCS*, vol. 2414, Aug. 2002, pp. 98–113.

[12] C. F. Lin and F. Y. S. Lin, "Efficient Estimation and Collision-Group-Based Anticollision Algorithms for Dynamic Frame-Slotted ALOHA in RFID Networks," *IEEE Trans. Automation Sci. and Eng.*, vol. 7, no. 4, Oct. 2010, pp. 840–48.

[13] H. Wu *et al.*, "Binary Tree Slotted ALOHA for Passive RFID Tag Anti-Collision," I*EEE Trans. Parallel and Distrib. Sys.*, vol. 24, no. 1, 2013, pp. 19–31.

Biographies

YEJUN HE [SM'09] (heyejun@126.com) received his PhD degree in Information and Communication Engineering from Huazhong University of Science and Technology (HUST) in 2005, MS degree in Communication and Information System from Wuhan University of Technology (WHUT) in 2002, and his BS degree from Huazhong University of Science and Technology in 1994. From Sept. 2005 to Mar. 2006, he was a Research Associate with the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University. From April 2006 to Mar. 2007, he was a Research Associate with the Department of Electronic Engineering, Faculty of Engineering, The Chinese University of Hong Kong. From July 2012 to August 2012, he was a Visiting Professor with Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. Dr. He is currently a Professor at Shenzhen University, China. His research interests include channel coding and modulation; MIMO-OFDM wireless communication; space-time processing; smart antennas; RFID and so on. Dr. He is a senior member of China Institute of Communications and China Institute of Electronics. He is also serving or has served as a reviewer/technical program committee member/session chair for various journals and conferences, including IEEE Transactions on Vehicular Technology, IEEE Communications Letters, International Journal of Communication Systems, IEEE VTC (2008-Spring, 2009-Spring, 2012-Fall), IEEE WCNC2012, IEEE WCNC2013 and so on. He acted as the Publicity Chair of IEEE PIMRC2012. He serves as an Associate Editor of the *Security and Communication Networks Journal* since 2012.

XIAOYE WANG (wangxiaoye2013@126.com) received her B.S. degree in electronic information engineering from Hunan University of Science and Engineering, China, in 2010 and is pursuing her M.S. degree at Shenzhen University, China. Her research interests include RFID and cognitive radio.